Name__________________________________

EET 1131 Lab #4
Introduction to Quartus II Software

Introduction
In the steps below you'll use Altera's Quartus II software to program a simple Boolean expression into a programmable logic device. This exercise will illustrate the steps of entering a design, simulating it, and burning it onto a programmable chip.

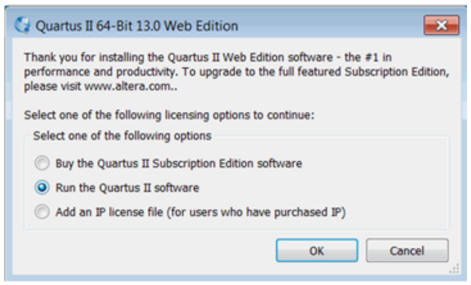
The expression that we'll use is $X = AB + \overline{C}D$. Here is the truth table for this expression. You'll need this later in the lab to verify that your chip is working correctly. (In another week or so you'll know how to create this truth table yourself from scratch.)

A	*B*	*C*	*D*	*X*
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Part 0. Starting Quartus II
The first time you start Quartus II, you'll a couple of dialog boxes that will never appear again after this first start-up. Follow the steps below to get Quartus II up and running.

Procedure
1. From your computer's **Start** menu, go into the **Altera** folder and start the program named **Quartus II 13.0 sp1**.

2. You should soon see the dialog box shown below. Select **Run the Quartus II software** and press **OK**.



Quartus II 64-Bit 13.0 Web Edition

Thank you for installing the Quartus II Web Edition software - the #1 in performance and productivity. To upgrade to the full featured Subscription Edition, please visit www.altera.com..

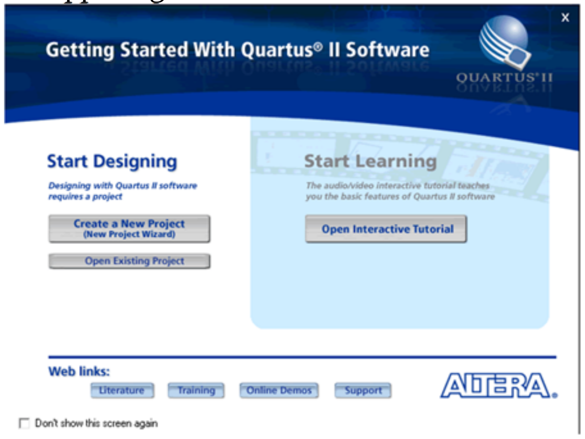
Select one of the following licensing options to continue:

Select one of the following options

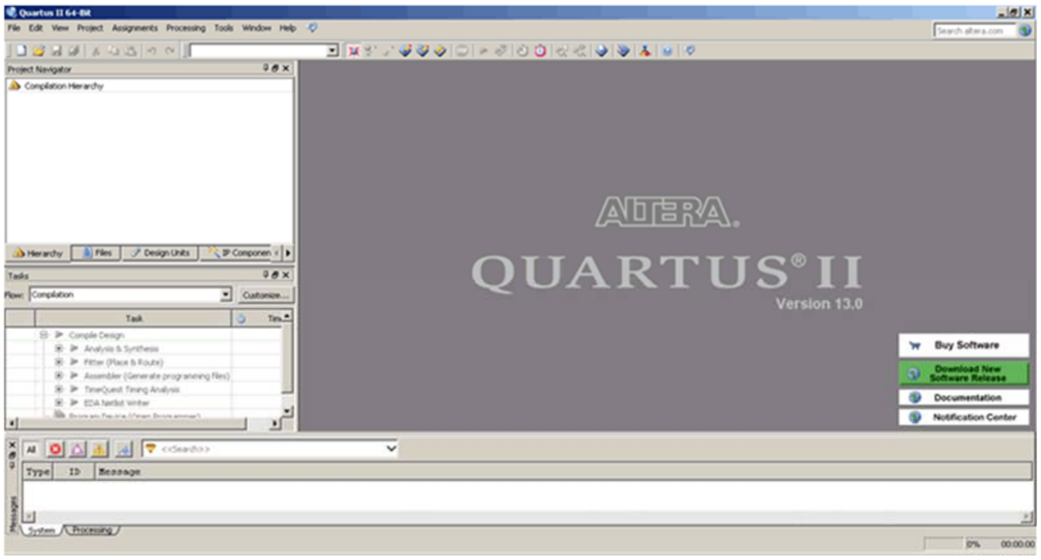
- Buy the Quartus II Subscription Edition software
- [✓ Run the Quartus II software]
- Add an IP license file (for users who have purchased IP)

OK Cancel

3. You should soon see the "Getting Started" screen shown below. I find this screen annoying and not very useful, so I recommend checking the box in the lower left corner that says "Don't show this screen again." Then close this screen by clicking the X in its upper right corner.


Getting Started With Quartus® II Software
QUARTUS®II
Start Designing
Designing with Quartus II software requires a project
Create a New Project (New Project Wizard)
Open Existing Project
Start Learning
The audio/video interactive tutorial teaches you the basic features of Quartus II software
Open Interactive Tutorial
Web links:
Literature
Training
Online Demos
Support
ALTERA.
Don't show this screen again

4. Now you're into Quartus II and your screen should look as shown below. (You might see a small "Tips and Tricks" window. I recommend closing it.) Maximize the window so that it fills your computer screen.



Part 1. Creating a Project

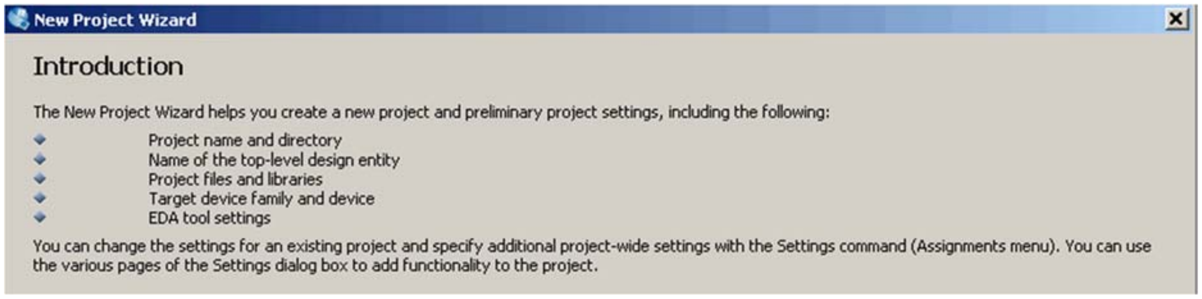
Any design task using Quartus II involves a large number of files. You create some of these files yourself, such as:

- A block diagram/schematic file, whose filename ends in **.bdf**.
- A vector waveform file, whose filename ends in **.vwf**.

In addition to the files that you create, Quartus automatically creates many others. All of these files are stored as a "project" inside a single folder on your computer. So when you start working on a new design, your first step will be to create a new project.

Procedure

1. Choose **File > New Project Wizard**. You'll see the Introduction screen shown below. Click the **Next** button.



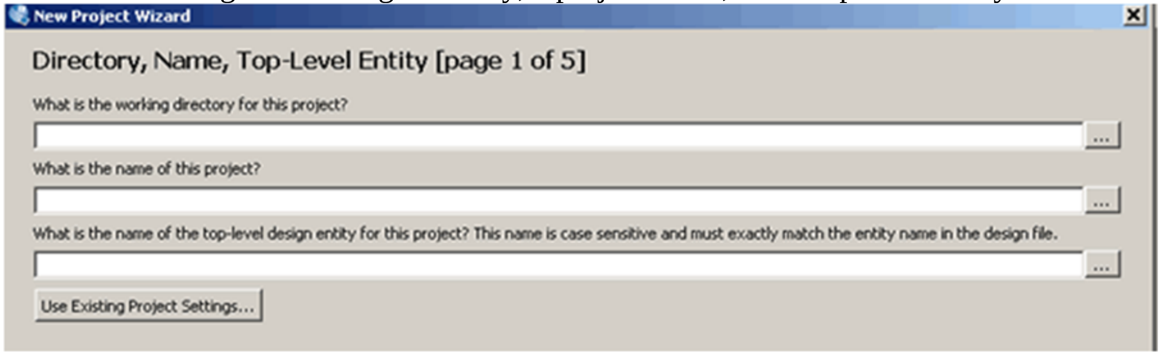
Introduction

The New Project Wizard helps you create a new project and preliminary project settings, including the following:

- Project name and directory
- Name of the top-level design entity
- Project files and libraries
- Target device family and device
- EDA tool settings

You can change the settings for an existing project and specify additional project-wide settings with the Settings command (Assignments menu). You can use the various pages of the Settings dialog box to add functionality to the project.

2. As shown below, you'll see **page 1** of the New Project Wizard, which asks you for three things: a working directory, a project name, and a top-level entity name.



Directory, Name, Top-Level Entity [page 1 of 5]

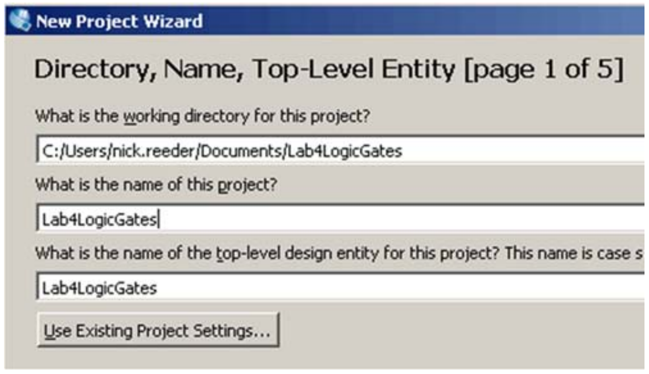
What is the working directory for this project?

What is the name of this project?

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.

3. For the *working directory*, click the button labeled **…**. A dialog box named **Select Folder** will open. Navigate to the **My Documents** folder and click the **Select Folder** button. You should now see **C:/Users/nick.reeder/Documents** (but with your name instead of mine) in the first box. At the end of this, type **/Lab4LogicGates**.

4. For the *name of the project*, type **Lab4LogicGates**. Notice that as you type this in for the project's name, it also appears in the third box, which is for the *name of the top-level design entity*. <u>You should always use the same name for your working directory, your project, and your top-level design entity</u>. So your filled-in dialog box will look like this:


New Project Wizard

Directory, Name, Top-Level Entity [page 1 of 5]

What is the working directory for this project?

C:/Users/nick.reeder/Documents/Lab4LogicGates

What is the name of this project?

Lab4LogicGates

What is the name of the top-level design entity for this project? This name is case s

Lab4LogicGates

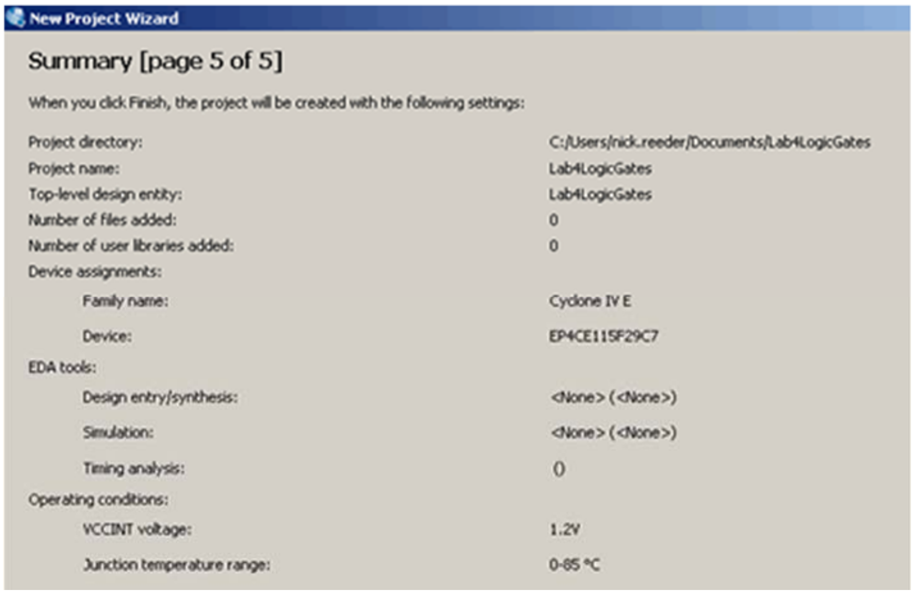
5. Press the **Next** button, and you'll see a message asking if you want to create the directory "C:/Users/nick.reeder/Documents/Lab4LogicGates." Click **Yes**.

6. You'll see **page 2** of the New Project Wizard, which lets you add files to your project. You don't have any files to add, so just press the **Next** button.

7. **Page 3** of the New Project Wizard will appear, asking you to specify the chip onto which you will burn your design. Fill in this dialog box as follows:
   - In the **Family** drop-down box, select **Cyclone IV E**.
   - Under **Available devices**, select **EP4CE115F29C7**, which is near the bottom of the list.
   - Press the **Next** button.

8. **Page 4** of the wizard will appear, letting you specify settings for any electronic-design automation tools (EDA tools) that you'll be using in addition to Quartus II. Since you won't use any other EDA tools, just press the **Next** button.

9. **Page 5** of the wizard will appear, asking you to confirm everything you've entered. It should look like this:


New Project Wizard

Summary [page 5 of 5]

When you click Finish, the project will be created with the following settings:

Project directory:	C:/Users/nick.reeder/Documents/Lab4LogicGates
Project name:	Lab4LogicGates
Top-level design entity:	Lab4LogicGates
Number of files added:	0
Number of user libraries added:	0
Device assignments:	
Family name:	Cyclone IV E
Device:	EP4CE115F29C7
EDA tools:	
Design entry/synthesis:	<None> (<None>)
Simulation:	<None> (<None>)
Timing analysis:	0
Operating conditions:	
VCCINT voltage:	1.2V
Junction temperature range:	0-85 °C

10. Press the **Finish** button.

Part 2. Creating a Block Diagram/Schematic File

There are two main ways to enter a design in Quartus II. The first way, which you'll learn now, is to draw a schematic diagram, much as you would in Multisim. When you do this, you save the diagram in a file called a **block diagram/schematic file**. The other way, which you'll learn later, is to type a description of your design using a special language called VHDL.

In this part of the lab you'll use the schematic-diagram method to implement the Boolean equation:

$$X = AB + \overline{C}D$$

Procedure

1. In the Quartus II menus, select **File > New**.

2. In the resulting dialog box, select **Block Diagram/Schematic File** and press **OK**. A new window named Block1.bdf will open.

3. Save this new file as part of your project by selecting **File > Save As…** from the menus and filling in the dialog box as follows:
    - Give this file a name of **Lab4LogicGates**. (<u>This file's name must be the same as your project's name</u>.)
    - Make sure that **Save as type** is set to **Block Diagram/Schematic File (*.bdf)**.
    - Make sure there's a checkmark in the **Add file to current project** box.
    - Then press the **Save** button.

You've now got an empty workspace in which you'll draw your schematic diagram. This is very much like drawing a schematic diagram in Multisim, although there are differences that you'll get used to as you practice.

4. Right-click your mouse anywhere in the empty workspace.

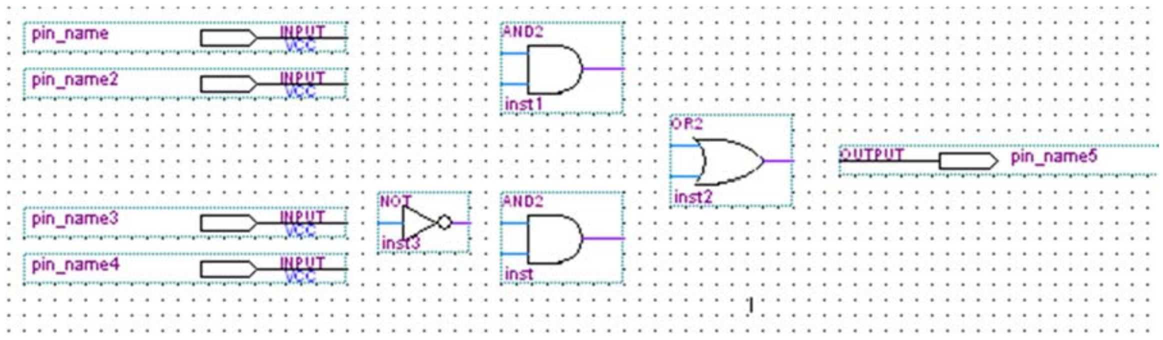
5. In the shortcut menu that appears, select **Insert > Symbol**.

6. In the **Name** box, type **and2**. (This is the name for a two-input AND gate.) Then press the **OK** button.

7. Move your mouse to wherever you'd like to place the AND gate, and press your left mouse button.

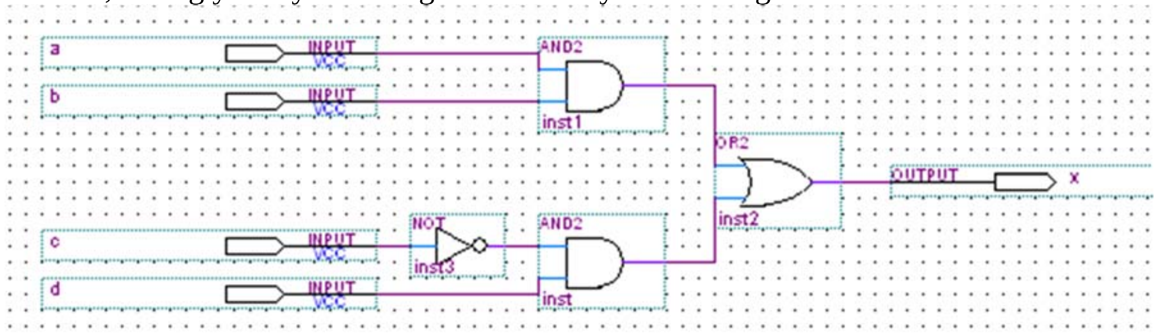
8. Repeat the steps above to place another two-input AND gate, a two-input OR gate (which is called **or2**), and an inverter (which is called **not**).

9. Next, follow these steps to place four input pins and one output pin:
    - Right-click in the empty workspace.
    - Select **Insert** > **Symbol**.
    - In the **Name** box, type **input**. Then press **OK**.
    - Place the input pin to the left of the gates.
    - Repeat for three more input pins and one output pin. Arrange the symbols so that your workspace looks like this:



10. To change the name of an input pin or output pin, simply double-click its current name. Do this now, renaming your input pins to **a**, **b**, **c**, and **d** (from top to bottom). Also rename your output pin to **x**.

11. Now you must wire these gates and pins to each other. This is similar to drawing wires in Multisim, but it's a little different. In Multisim, to draw a wire, you click and release the mouse button, and then move the mouse. In Quartus II, you must hold the mouse button down as you move the mouse; **don't release the mouse button** until you've reached the point that you want the wire to go to. Do this now, wiring your symbols together so that your drawing looks like this:



12. Save your file by selecting **File** > **Save** from the menus.

Part 3. Compiling the Project

Next you'll compile your design. In this step, Quartus II does a number of things. It checks your design for errors. It also generates the code needed to program your design into the specific chip that you selected earlier.

Procedure

1. Select **Processing > Start Compilation** from the menus. Compilation will take a minute or so, during which you should see a spinning clock circle in the screen's lower right corner. Eventually, you should see a message telling you that full compilation was successful (with some number of warnings). Warnings are okay; but if you have any errors, then you've done something wrong above and need to fix it before you proceed.

2. Press the **OK** button.

Part 4. Simulating Your Design

At this point you could download your design onto the chip. But first, you should simulate it in software to make sure that your design will produce the correct output(s). In many cases you'll catch mistakes at this point, which you can correct before you burn the design onto the chip. This will save you time, since Quartus II can test your design in the "virtual world" faster than you can test it in the real world.

To simulate your design, you must first create a **vector waveform file (VWF)**.

Procedure

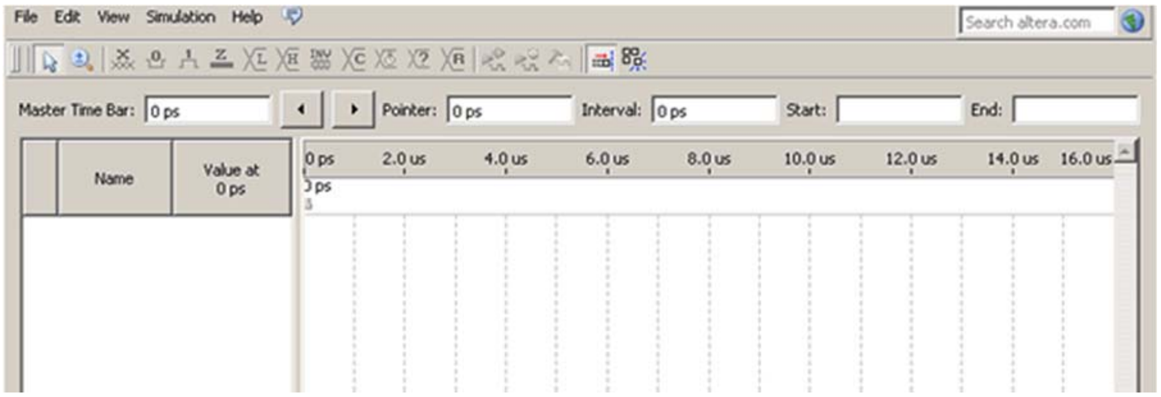
1. Select **File > New** from the menus. In the resulting dialog box, select **University Program VWF**, and then press **OK.** A new file will open in a new window named **Simulation Waveform Editor**.

2. Save this new file as part of your project by selecting **File > Save As…** from the menus and filling in the dialog box as follows:
   - Give this file a name of **Lab4LogicGates**.
   - Make sure that **Save as type** is set to **University Program VWF (*.vwf)**.
   - Make sure there's a checkmark in the **Add file to current project** box.
   - Then press the **Save** button.

   Next you must set up the timing for your simulation, which involves telling Quartus II two things:
   - How often it should change the input values. (This is called the **grid size**.) In the steps below, we'll set the grid size to 1 µs.
   - How long the simulation should run. (This is called the **end time**.) In our case, we have four input variables, and we want to simulate all possible combinations of these four inputs. That's a total of 16 possible combinations. Therefore, since we're allowing 1 µs for each input combination, our end time will be 16 µs.

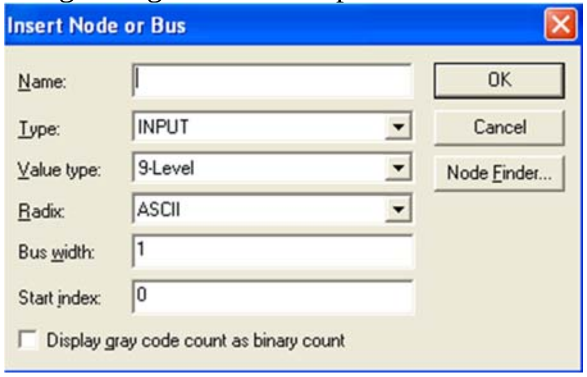
3. Select **Edit > Grid Size…** from the menus. In the resulting dialog box, set **Period** to 1 and set the unit to **us**. Then press **OK**.

4. Select **Edit > Set End Time…** from the menus. In the resulting dialog box, set **End Time** to 16 and set the unit to **us**. Then press **OK**. Your window should now look like this:



Next you need to add the input and output variables that you want to simulate.

5. Near the left edge of the window, double-click in the blank area under the word **Name**. The following dialog box should open:



Name:
Type: INPUT
Value type: 9-Level
Radix: ASCII
Bus width: 1
Start index: 0
☐ Display gray code count as binary count

6. Type **a** in the **Name:** box and press **OK**.

7. Repeat the previous two steps several times to add **b**, **c**, **d**, and **x** in the **Name** column underneath **a**.

Next, how do you tell Quartus II to step through all of the possible input combinations for *a*, *b*, *c*, and *d*? The easiest way is to tell Quartus II to treat each of these inputs as a clock, with each input having a period half as long as the input before it. In reality, we don't plan to use these four inputs as clock inputs. But this is an easy way to get Quartus II to step through all possible input combinations.

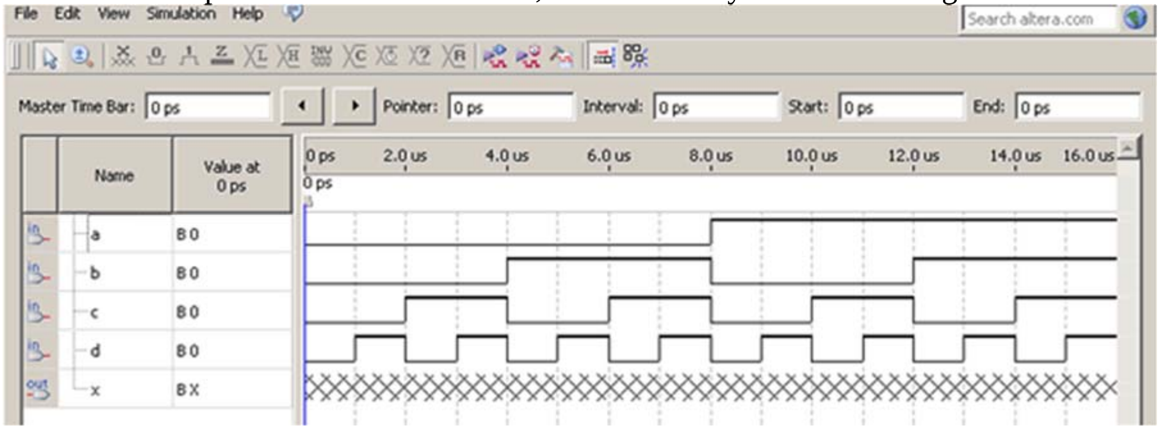
8. Highlight the variable name *a* in the **Name** column by clicking it.

9. Select **Edit > Value > Overwrite Clock** from the menus. In the resulting dialog box, set **Period** to 16 and set the unit to **us**. Then press **OK**.

10. Similarly, select *b* and make it a clock with a period of 8 µs.

11. Then select *c* and make it a clock with a period of 4 µs.

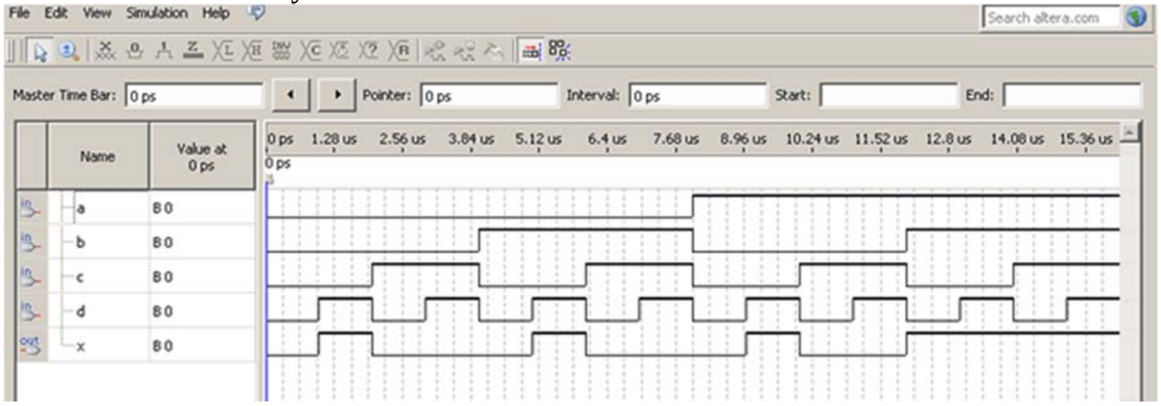
12. Finally, select *d* and make it a clock with a period of 2 µs. At this point your timing diagram should look as shown below. Notice that *a*, *b*, *c*, and *d* step through all possible combinations. Since we haven't run the simulation yet, the value of output variable *x* is unknown, as indicated by the crossed diagonal lines.



13. Save your file by selecting **File > Save** from the menus.

Now that we've done the work of setting up the combinations of input variables to be simulated, it's time to tell Quartus II to perform the simulation.

14. Select **Simulation > Run Functional Simulation** from the menus. A new Simulation Flow window will open and quickly display some messages that you don't need to worry about. After a few seconds, another new window will open with the results of your simulation. It should look like this:


File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 0 ps
Interval: 0 ps
Start:
End:
Name
Value at 0 ps
a B0
b B0
c B0
d B0
x B0
0 ps 1.28 us 2.56 us 3.84 us 5.12 us 6.4 us 7.68 us 8.96 us 10.24 us 11.52 us 12.8 us 14.08 us 15.36 us

15. Using the truth table from the beginning of this lab, verify that the output waveform is correct for $x = ab + \overline{c}d$. Then show me your simulation results.

____

Part 5. Assigning Input and Output Pin Numbers

So far you've simulated your design, and you've also (in Part 1) told Quartus II that you plan to download this design onto an Altera Cyclone IV EP4CE115F29C7 chip. Before you actually download the design to the chip, you must tell Quartus II which pins on the chip to use for your circuit's four inputs (*a*, *b*, *c*, *d*) and your circuit's one output (*x*).

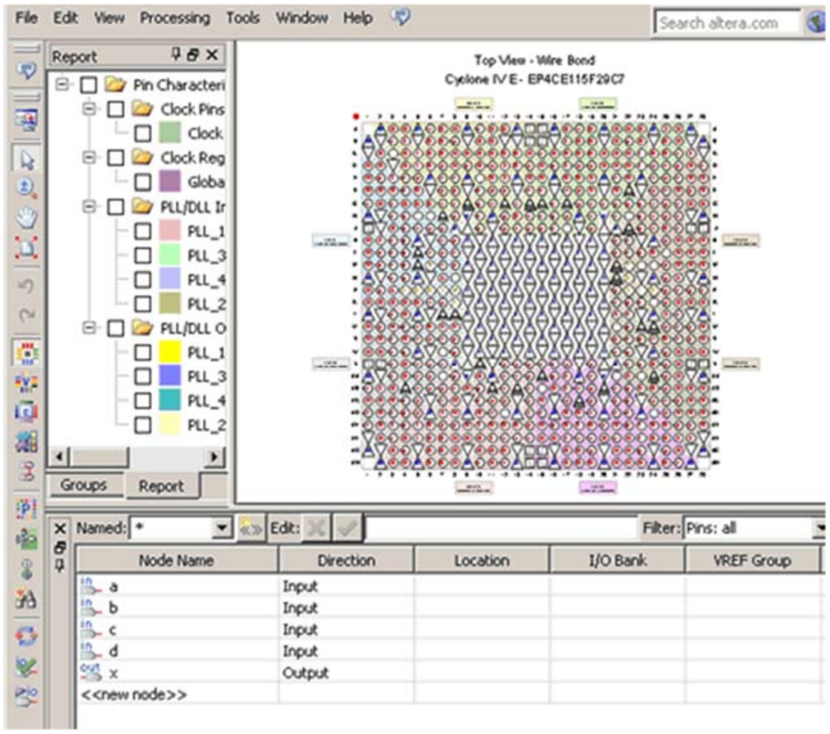
The Altera DE2-115 board contains many switches and LEDs. Locate the bank of slide switches along the board's lower edge. Notice that these switches are labeled SW0 through SW17. You'll use four of these switches to enter the values of your inputs *a*, *b*, *c*, and *d*. Also locate the bank of red LEDs just above the slide switches. You'll use one of these LEDs to display your output *x*.

All of the switches and LEDs are hard-wired to certain pins on the Cyclone IV chip. The DE2-115 User Manual on the course website tells you which devices are connected to which of the chip's pins. Look now the tables on pages 36-37 of the User Manual. Note that, for example, the rightmost switch, SW[0], is connected to pin AB28 on the chip. Using these tables, fill in the last column of the table below, in which I have selected certain switches and LEDs for your circuit's input and outputs:

Variable	Switch or LED	Pin number
a	SW17	
b	SW16	
c	SW15	
d	SW14	
x	LEDR[17]	

Procedure

1. In Quartus II, select **Assignments > Pin Planner** from the menus. You'll see the Pin Planner window shown below. Notice that near the bottom of this window, your input and output names are listed in the **Node Name** column.



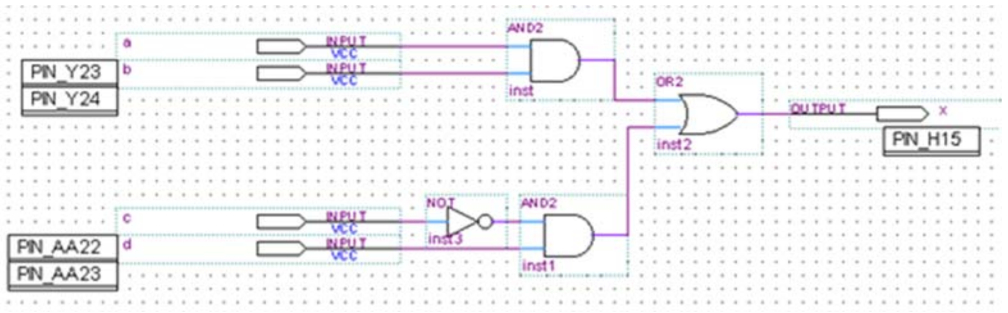
Node Name	Direction	Location	I/O Bank	VREF Group
a	Input			
b	Input			
c	Input			
d	Input			
x	Output			
<<new node>>				

2. Double-click in the **Location** column of the first row (the row whose Node Name is **a**). Then type **Y23** and press your keyboard's **Enter** key.

3. Repeat the previous two steps several times, entering the pin numbers for your other inputs and outputs.

4. After you've entered the five pins numbers, close the Pin Planner window. Then select **File > Save Project** from Quartus II's menus.

5. Your .bdf file (which should still be open) now displays the pin numbers that you've assigned to the inputs and outputs, as shown below:



6. Compile your design again by selecting **Processing > Start Compilation** from the menus. After a delay, you should get a message telling you that compilation was successful (with some warnings). Press the **OK** button.

Part 6. Downloading Your Design to the FPGA

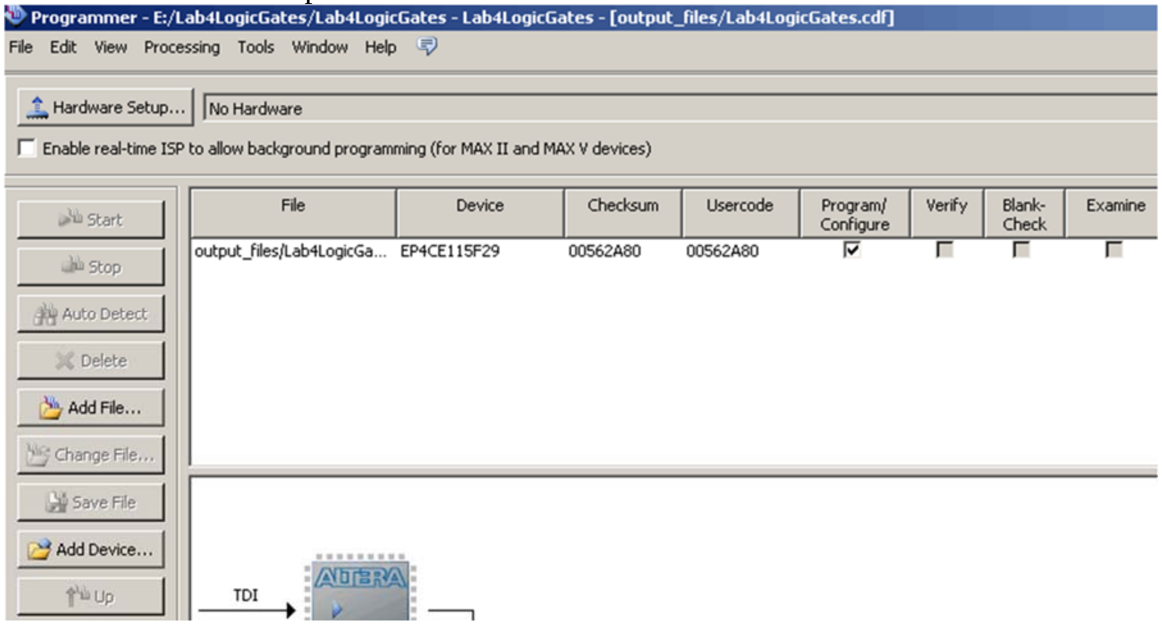
Now you're ready to program the chip!

Procedure

1. Plug one end of a USB cable into your computer's USB port, and the other end into the leftmost USB port on the top edge of the DE2-115 board.

2. Make sure that the power cord is plugged into a wall socket, and the other end is plugged into the DE2-115 board. Press the DE2-115's red ON/OFF switch. The DE2-115's LEDs should light up.

3. In Quartus II, select **Tools > Programmer** from the menus. The dialog box shown below will open.



Programmer - E:/Lab4LogicGates/Lab4LogicGates - Lab4LogicGates - [output_files/Lab4LogicGates.cdf]

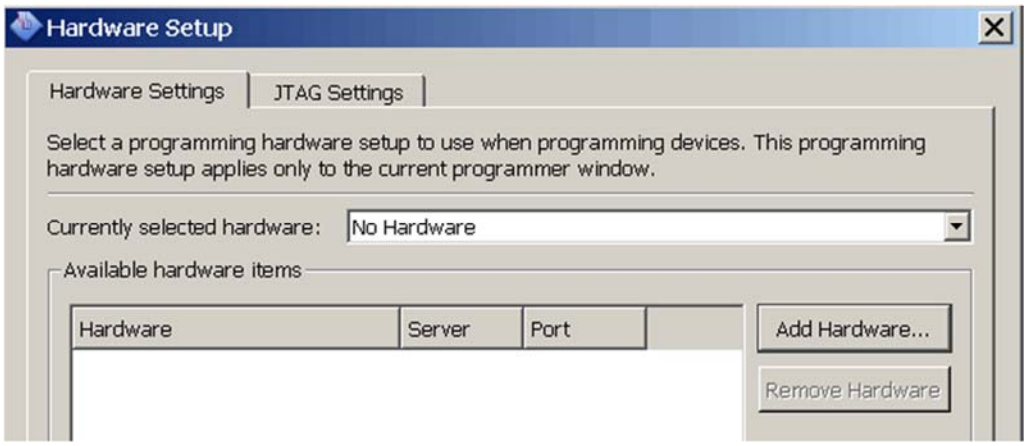
Hardware Setup... No Hardware

☐ Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
output_files/Lab4LogicGa...	EP4CE115F29	00562A80	00562A80	✓	☐	☐	☐

In the picture above, notice the words **No Hardware** to the right of the **Hardware Setup** button near the top of the picture. The first time you use your computer to program the DE2-115, your screen will probably also say **No Hardware**, and so you must complete the next few steps to configure the hardware connection. But when you're working on future projects, you should see the words **USB-Blaster [USB-0]** instead of **No Hardware**. Then you can skip these next few steps.

4. Press the **Hardware Setup…** button. The **Hardware Setup** dialog box shown below will open.


Hardware Setup

Hardware Settings | JTAG Settings

Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

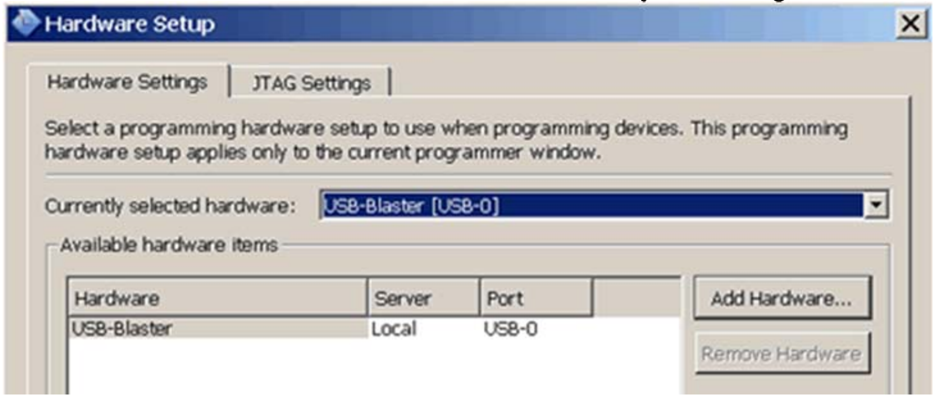
Currently selected hardware: No Hardware

Available hardware items

Hardware	Server	Port

5. Press the **Add Hardware…** button. An **Add Hardware** dialog box will open.
   - For **Hardware type**, select **USB-Blaster**.
   - For **Port**, select **USB-0.**
   - Press **OK**.

6. Back in the **Hardware Setup** window, use the drop-down box to select **USB-Blaster [USB-0]** instead of **No Hardware**, so that your dialog box looks like this:


Hardware Setup

Hardware Settings | JTAG Settings

Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

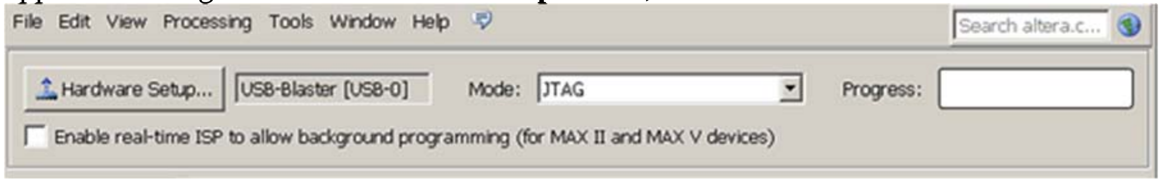
Currently selected hardware: USB-Blaster [USB-0]

Available hardware items

Hardware	Server	Port
USB-Blaster	Local	USB-0

7. Press the **Close** button.

8. Back in the main dialog box, verify that the words **USB-Blaster [USB-0]** now appear to the right of the **Hardware Setup** button, as shown below.



USB-Blaster [USB-0]   Mode: JTAG   Progress:

Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

9. Make sure there's a check in the checkbox under the words **Program/Configure** next to your output file's name. Then press the **Start** button. You should see a progress bar telling you the status of the download. The download should complete with no error messages.

Part 7. Testing the Programmed FPGA

The first page of this lab shows the truth table for the expression that you've just burned onto the chip. Now let's verify that the chip behaves as expected.

Procedure

1. Examine the slide switches on the DE2-115. Each one produces a LOW (or 0) when it's slid toward you, and a HIGH (or 1) when it's slid away from you.

2. **DO NOT** use a pencil to move the slide switches, since graphite from the pencil can foul the switch contacts and cause the switch to fail. Use your fingertip or a small screwdriver instead.

3. Use the slide switches to run through all possible combinations of the input variables. Using a straight-edge, record your results in a truth table below.

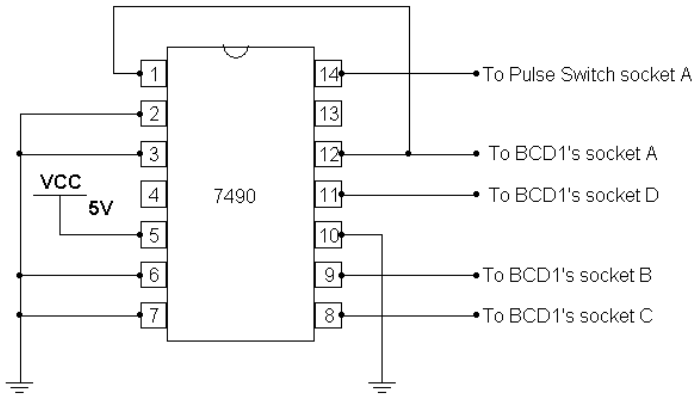
4. After you've verified that your truth table matches the one on this lab's first page, show me your working circuit.

____

5. Close this project by selecting **File > Close Project** from the menus. For future use, copy the Lab4LogicGates folder from your computer to your flash drive.

Part 8. Programming a Counter Circuit

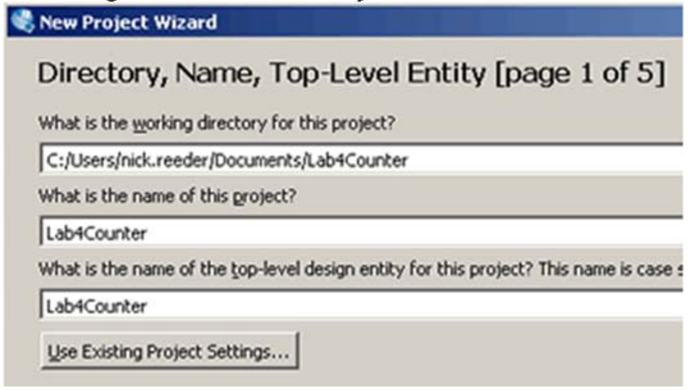
You've learned how to use Quartus II to program the Cyclone FPGA chip to implement a simple circuit containing a few gates. You can also design more complicated circuits that contain many 74XX chips, and **then burn your entire design onto a single Cyclone chip**. Let's keep things simple for now by just programming the Cyclone chip to behave like the 7490 counter circuit that you built in Lab #1, which is shown below.



This will require you to run through most of the steps that you performed above, with the changes noted below.

Procedure

1. **Create a New Project**: Following the steps that you performed above in Part 1 of this lab, create a new project called **Lab4Counter** in a new folder named **Lab4Counter**. Page 1 in the New Project Wizard should look like this:



Directory, Name, Top-Level Entity [page 1 of 5]

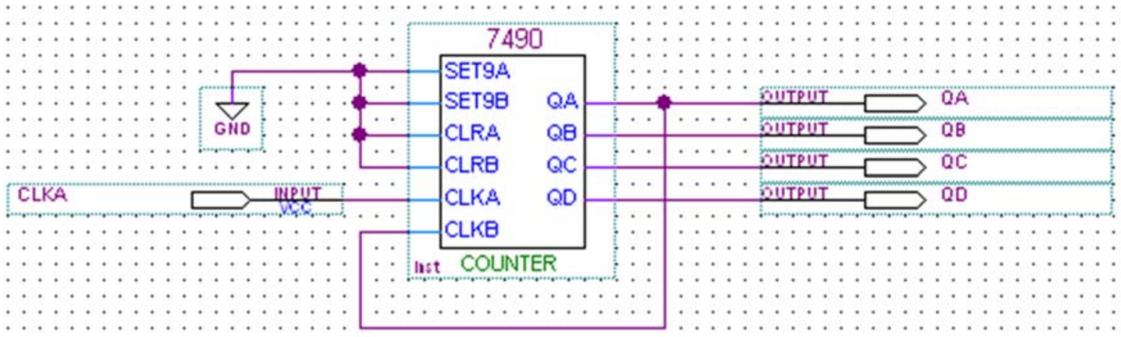
What is the working directory for this project?
C:/Users/nick.reeder/Documents/Lab4Counter

What is the name of this project?
Lab4Counter

What is the name of the top-level design entity for this project? This name is case s
Lab4Counter

2. **Create the Block Diagram/Schematic File**: Following the steps that you performed above in Part 2, create a new .bdf file. In your .bdf file, build the schematic shown below. Two new symbols that you'll need for your design are the ones named **7490** and **GND**. As shown below, name your input pin **CLKA,**

and name your output pins **QA**, **QB**, **QC**, and **QD**. Save this file under the name **Lab4Counter.bdf**.



3. **Compile the Project**: Compile your design as you did above in Part 3. You should get some warnings, but no errors.

4. **Simulate Your Design**: Following the steps that you performed above in Part 4, create a .vwf file that displays your circuit's one input and four outputs. Set the grid size to 1 μs and the end time to 16 μs. Also configure the CLKA input to behave as an Overwrite Clock with a time period of 1 μs. When you run the simulation, you should find that the four output bits count in binary from 0000_2 to 1001_2, and then repeat. Show me your simulation results.

____

5. **Assign Input and Output Pin Numbers**: Referring to the DE2-115's User Manual, find out which pin number is connected to the leftmost push-button. Following the steps that you performed above in Part 5, assign this pin number to your circuit's input pin. Also refer to the manual to find out which pin numbers are connected to the four rightmost green LEDs. Assign these pin numbers to your circuit's output pins, with output QA connected to the rightmost LED, output QB connected to the next LED, and so on. Then recompile your design.

6. **Download Your Design to the FPGA**: Following the steps that you performed above in Part 6 (but this time you should be able to skip the steps to set up the USB-Blaster), burn your design onto the Cyclone chip.

7. **Test the Programmed FPGA**: You should find that the four LEDs count up in binary as you press the pulse switch. Show me your working circuit.

____

6. Close this project by selecting **File > Close Project** from the menus. For future use, copy the Lab4Counter folder from your computer to your flash drive.

Part 9. A Blinking LED

As you know from previous labs, you can use the red trainer's built-in function generator to produce a signal that switches between HIGH and LOW at a frequency of your choice.

The DE2-115 doesn't have a function generator with adjustable knobs, but it does gives you a way to produce signals that switch between HIGH and LOW. Let's see how to do it.

First, let's do some calculations. One of the Cyclone's input pins carries a clock signal whose frequency is 50 MHz. We'll need to reduce that to a low enough frequency that we can see with our eyes. To do this we'll use a circuit called a **frequency divider**. As its name suggests, if we give it an input signal of a certain frequency, it will produce an output signal with a lower frequency. If the input frequency is 50 MHz and the frequency divider divides the frequency by 16, what will the output frequency be?

____

Now if we divide that frequency by 16 again, what will the frequency be?

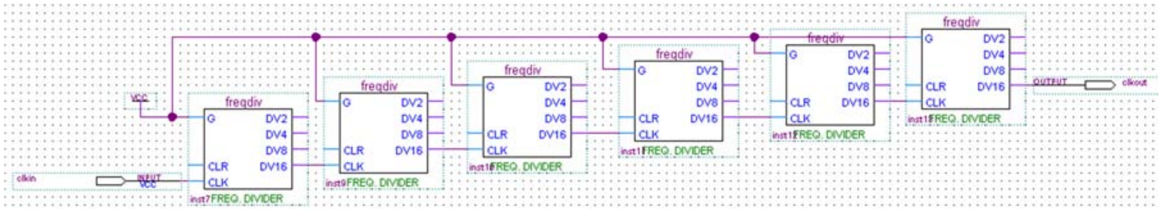
____

Now if we divide that frequency by 16 **four more times**, what will the frequency be?

____

1. **Create a New Project**: Following the steps that you performed above in Part 1 of this lab, create a new project called **Lab4BlinkingLED** in a new folder named **Lab4BlinkingLED**.

2. **Create the Block Diagram/Schematic File**: Following the steps that you performed above in Part 2, create a new .bdf file. In your .bdf file, build the schematic shown below. Two new symbols that you'll need for your design are the ones named **freqdiv** and **VCC**. Name your input pin **clkin,** and name your output pin **clkout**. Save this file under the name **Lab4BlinkingLED.bdf**.



3. **Compile the Project**: Compile your design as you did above in Part 3. You should get some warnings, but no errors.

4. **Assign Input and Output Pin Numbers**: Referring to the DE2-115's User Manual, find out which pin number is connected to a 50 MHz clock signal. (Hint: Do a search for the name **CLOCK_50**.) Following the steps that you performed above in Part 5, assign this pin number to your circuit's input pin. Also assign a pin number connected to any LED to your circuit's output pin. Then recompile your design.

5. **Download Your Design to the FPGA**: Following the steps that you performed above in Part 6, burn your design onto the Cyclone chip.

6. **Test the Programmed FPGA**: You should find that the LED blinks on and off a few times each second. Show me your working circuit.

____

7. Close this project by selecting **File > Close Project** from the menus. For future use, copy the Lab4BlinkingLED folder from your computer to your flash drive.

Part 10. A Brief Introduction to VHDL

Up to now, you've been using the schematic-entry method to enter your designs. For complex designs, the text-entry method can be more efficient. When using this method, you type a description of your design using a hardware description language. One popular hardware description language is VHDL. Let's use VHDL to implement the same Boolean expression that you implemented above using the schematic-entry method:

$$X = AB + \overline{C}D$$

Procedure

1. **Create a New Project**: Following the steps that you performed in Part 1 above, create a new project called **Lab4VHDL** in a new folder called **Lab4VHDL**.

2. **Create a VHDL File**:
   a. In the Quartus II menus, select **File** > **New**.
   b. In the resulting dialog box, select **VHDL File** and press **OK**. A text-editor window will open. Type the following VHDL program in this window:

```
ENTITY Lab4VHDL IS
     PORT(
          a, b, c, d: IN BIT;
          x:          OUT BIT);
END Lab4VHDL;

ARCHITECTURE arc OF Lab4VHDL IS
     BEGIN
          x <= (a AND b) OR (NOT c AND d);
END arc;
```

   c. To save your file, select **File** > **Save** from the menus and fill in the dialog box as shown below.
      i. Give this file a name of **Lab4VHDL**.
      ii. Make sure that **Save as type** is set to **VHDL File (*.vhd; *.vhdl)**.
      iii. Make sure there's a checkmark in the **Add file to current project** box.
      iv. Press the **Save** button.

3. **Compile**: Following the steps that you performed in Part 3 above, compile your design. You may get some warnings, but you shouldn't get any errors.

4. **Simulate**: Following the steps that you performed in Part 4 above, create a .vwf file. Set up your .vwf file to step the input variables through all of their possible combinations, as you did earlier. Then run the simulation and make sure that your output variable (*x*) has the right value for each input combination. Show me your simulation results.

____

5. **Assign Pin Numbers**: Using the DE2-115 User Manual, fill in the last column of the table below, in which I have selected certain switches and LEDs for your circuit's input and outputs:

Variable	Switch or LED	Pin number
a	SW3	
b	SW2	
c	SW1	
d	SW0	
x	LEDR[0]	

   Following the steps that you performed in Part 5 above, assign these pin numbers and then recompile your design.

6. **Download**: Following the steps that you performed in Part 6 above, download your design to the Cyclone chip.

7. **Test**: After testing your circuit's operation, show me your working circuit.

____

8. Close this project by selecting **File > Close Project** from the menus. For future use, copy the Lab4VHDL folder from your computer to your flash drive.